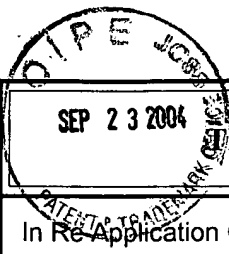


IFW Af



SEP 23 2004

TRANSMITTAL OF APPEAL BRIEF (Large Entity)

Docket No:
NCR.0026US

In Re Application Of: **Donald R. Peterson et al.**

Application No.	Filing Date	Examiner	Customer No.	Group Art Unit	Confirmation No.
09/784,392	February 15, 2001	Chongshan Chen	21906	2172	2999

Invention:


OPTIMIZED END TRANSACTION PROCESSING

COMMISSIONER FOR PATENTS:

Transmitted herewith in triplicate is the Appeal Brief in this application, with respect to the Notice of Appeal filed on

The fee for filing this Appeal Brief is: **\$330.00**

- ☐ A check in the amount of the fee is enclosed.
- ☐ The Director has already been authorized to charge fees in this application to a Deposit Account.
- ☒ The Director is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No. **50-1673(9417)**

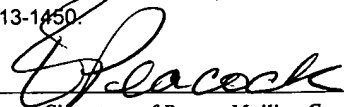


Signature

Dated: **September 20, 2004**

**Dan C. Hu, Reg. No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, Texas 77024
(713)468-8880 [Phone]
(713)468-8883 [Fax]**

I certify that this document and fee is being deposited on **September 20, 2004** with the U.S. Postal Service as first class mail under 37 C.F.R. 1.8 and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.



Signature of Person Mailing Correspondence

Ellen Peacock

Typed or Printed Name of Person Mailing Correspondence

cc:



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Applicant:	§	
Donald R. Peterson et al.	§	Art Unit: 2172
	§	
Serial No.: 09/784,392	§	Examiner: Chongshan Chen
	§	
Filed: February 15, 2001	§	Conf. No.: 2999
	§	
For: OPTIMIZED END	§	Atty Docket: 9417 (NCR.0026US)
TRANSACTION PROCESSING	§	

Mail Stop **Appeal Brief-Patents**
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 1-43 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is the NCR Corporation by virtue of the assignment recorded at reel/frame 11603/0862.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 1-43 have been finally rejected and are the subject of this appeal.

09/24/2004 SSESHE1 00000014 501673 09784392

01 FC:1402 330.00 DA

IV. STATUS OF AMENDMENTS

No amendments have been submitted after final rejection.

V. SUMMARY OF THE INVENTION

Claim 1 recites a method of performing a transaction in a database system that comprises receiving a transaction to be performed, where the transaction is processed by a plurality of access modules. The method further comprises performing a flush of a transaction log from volatile storage to non-volatile storage by each access module before an end transaction procedure.

Independent claim 17 recites a database system having a plurality of storage media, the storage media comprising persistent storage; a volatile storage; and a plurality of access modules, where each access module is coupled to one of the plurality of storage media. Each of the access modules is adapted to flush a transaction log from the volatile storage to the persistent storage before performing an end transaction procedure.

Independent claim 21 recites an article comprising a medium storing instructions for enabling a processor-based system to receive a transaction to be performed, wherein the transaction is processed by a plurality of access modules; determine that a last step of the transaction involves the plurality of access models, where the last step is performed before an end transaction procedure; and flush a transaction log from a volatile storage to a non-volatile while the last step is performed by the plurality of access modules.

Independent claim 24 recites a method of performing a transaction in a database system that includes receiving a transaction to be performed on plural access modules in the database system; maintaining a log in volatile storage to track operations performed in the transaction; and writing the log to persistent storage before start of an end transaction procedure.

Independent claim 28 recites a database system having storage media comprising persistent storage; volatile storage; access modules coupled to the storage media; and a parsing engine coupled to the access modules, the parsing engine adapted to perform one of: (a) providing a directive with a message to perform a last step of a transaction and communicating the directive to the access modules, each access module responsive to the directive to perform a transaction log flush from the volatile storage to the persistent storage before performance an end transaction procedure; and (b) determine if each of the access modules has performed a transaction log flush before start of the end transaction procedure. The parsing engine is also adapted to avoid sending a broadcast directive to the access modules to cause performance of a transaction log flush during the end transaction procedure.

According to some embodiments, a database system 100 (Fig. 1 of the specification) includes multiple access modules 20. Specification, page 5, lines 1-18. A transaction received by a parsing engine is converted into steps that are performed by access modules 20. Specification, page 6, lines 13-28. For each step of a transaction, one or more entries are made in a transaction log, with each access module including its own transaction log. Specification, page 11, lines 14-18. UNDO and REDO entries are recorded by each access module in its transaction log, with UNDO and REDO entries enabling a transaction that was interrupted to be completed, either in the forward (REDO) or backward (UNDO) direction, as appropriate. Specification, page 7, lines 19-22.

An end transaction procedure is invoked in response to an end transaction directive (for an explicit transaction) or started automatically (for an implicit transaction). Specification, page 8. lines 18-31. Implicit transactions are distinguishable from explicit transactions in that the parsing engine knows all the steps that make up the implicit transaction before processing of the

transaction begins. Specification, page 9 lines 14-16. End transaction processing ensures that where a failure disrupts a transaction, the transaction nevertheless may be either re-implemented or rolled back in its entirety. Conventionally, end transaction processing involves multiple phases, including the flushing of a transaction log from volatile storage to stable storage. Specification, page 9, line 28-page 10, line 4.

However, according to some embodiments of the invention, to improve efficiency, flushing of a transaction log is moved from the end transaction procedure to the last step of a data access transaction. Specification, page 11, lines 25-30. A beneficial effect of optimizing the transaction log flushing is that the broadcast of a transaction log flush directive, the performance of a transaction log flush, and a “last done” coordination step during the end transaction procedure can be avoided or eliminated to enhance efficiency and to improve overall system performance. Specification, page 12, lines 1-6. One example embodiment of the transaction log flush prior to the end transaction procedure is depicted in Figure 6. Specification, page 12, line 16-page 13, line 21. The transaction flushing prior to end transaction procedure can also be performed for explicit transactions. Specification, page 13, lines 22-28.

Simplification of the end transaction procedure, which avoids the flushing of the transaction log during the end transaction procedure, is further described in the specification at page 14, line 1-page 15, line 15.

Independent claim 10 recites a method performing an end transaction procedure in a database system that includes a first access module in the database system writing an end transaction indication to a first transaction log portion, where the first access module is part of a cluster of access modules; and the first access module sending an end transaction directive to a

fallback module associated with the first access module, the fallback module being part of the cluster.

The subject matter of claim 10 is supported by a fallback feature of the database system described in the specification. Fallback is a redundancy operation in which a copy of a data portion, such a row of a table, is automatically stored on a different access module than where the original of the data portion is stored (on a primary access module). Specification, page 15, lines 17-24.

According to one embodiment, a cluster is a group of access modules that act as a single fallback unit. Specification, page 15, lines 25-31. Thus, a fallback module is an access module, different from a primary access module, for storing a copy of a data portion to enable recovery of the system in case of failure of the primary access module. Specification, page 16, lines 1-9.

As a optimization, instead of broadcasting an end transaction directive to all access modules, only one of the access modules (the “last done” access module) writes the end transaction directive to its transaction log. The “last done” access module then flushes its transaction log, and the “last done” access module sends the end transaction directive to its fallback access module. In response, the fallback access module also writes the directive to its transaction log and flushes the transaction log to stable storage. Specification, page 17, lines 4-19. Thus, the end transaction directive is sent to just two access modules and to their transaction logs in the same cluster: the “last done” access module and its fallback access module. Because the flush is occurring on two, rather than all, transaction logs, the need to perform “last done” coordination is avoided. Specification, page 17, lines 20-26.

VI. GROUNDS FOR REJECTION

- A. Claims 1-9 and 17-43 stand rejected under 35 U.S.C. § 103 over Tada in view of Bohannon.**
- B. Claim 10-16 stand rejected under 35 U.S.C. § 103 over Tada alone.**

VII. ARGUMENT

- A. Claims 1-9 and 17-43 stand rejected under 35 U.S.C. § 103 over Tada in view of Bohannon.**

- 1. Claims 1, 2, 8, 17, 18, 20, 24-27, 29, and 34-37.**

The examiner has failed to establish a *prima facie* case of obviousness against the claims for at least the following reasons: (1) the references when combined do not teach or suggest *all* elements of claimed subject matter; and (2) there existed no motivation or suggestion to combine teachings of Tada and Bohannon. See M.P.E.P. § 2143 (8th ed., Rev. 2), at 2100-129.

Contrary to the Examiner's assertion, Tada does not disclose the flushing of a transaction log from a volatile storage to non-volatile storage by each access module before an end transaction procedure. As recited in claim 1, flushing of the transaction log to non-volatile storage occurs before an end transaction procedure. This clearly is not disclosed by Tada, contrary to the assertion made in the Office Action. Because the Examiner has mis-applied Tada against an element of claim 1, the obviousness rejection is defective on at least this ground.

The Examiner cited to Figure 5 and the passage at column 10, lines 33-34, of Tada as disclosing the performing of a flush of a transaction log. See 4/5/2004 Office Action at 4. The cited passage refers to the transfer of log data to a log data buffer 132 on a main storage unit 101. Tada, 10:33-34. The main storage unit "is formed of a volatile random-access memory (RAM)." Tada, 8:13-14 (emphasis added). The log data buffer 132 (in addition to volatile HLF buffers

114) are part of the volatile main storage unit 101. Tada, 8:5-9. Thus, the transfer of the log data to log data buffer 132 is a transfer of data to volatile storage. Therefore, the cited passage in column 10 of Tada does not disclose the performance of flushing of a transaction log from volatile storage to non-volatile storage.

In Tada, the transfer of log data to HLF buffer 112 in non-volatile memory 103 does not occur until after the issuance of a transaction-end instruction. Tada, 10:43-44, 11:30-33. Therefore, the transfer of log data from volatile storage to non-volatile storage as performed in Tada is after (not before) a transaction-end instruction has occurred--in other words, the transfer of log data from volatile storage to non-volatile storage is part of an end transaction procedure, not before an end transaction procedure, as recited in claim 1.

The Examiner further stated that "it would have been obvious to one of ordinary skill in the art at the time the invention was made to flush the log to non-volatile storage so that the data will not be lost when the system goes down." 4/5/2004 Office Action at 2, 4. The Examiner also referred to Bohannon as teaching the flushing of the transaction log to non-volatile storage. 4/5/2004 Office Action at 2. While it may be known to flush a transaction log to non-volatile storage, there is no teaching or suggestion anywhere within any of the cited references of flushing the transaction log to non-volatile storage before an end transaction procedure. In fact, the opposite is suggested by Tada, which teaches writing of log data to non-volatile historical log files (HLF) in step S09 in Figure 5 of Tada. Step S09 is performed after the system has issued a transaction-end macro (step S06 in Figure 5 of Tada). See Tada, 10:43-44. In other words, in Tada, the writing of a transaction log from volatile storage to non-volatile storage occurs *during* an end transaction procedure, not *before* an end transaction procedure. Performing a flush of a transaction log to non-volatile storage before an end transaction procedure enables more efficient

processing, as discussed in the present application. One of the benefits achieved by the present application is, in certain cases, flushing of a transaction log can be avoided during an end transaction procedure. See Specification, page 13, lines 4-10. As a result, end transaction processing is made more efficient. The ability to perform a flush of a transaction log from volatile storage to non-volatile storage before an end transaction procedure is clearly not even remotely suggested by Tada.

Recognizing the defective logic of the obviousness rejection made in the final Office Action, the Examiner in the Advisory Action appeared to have shifted his position to indicate now that setting the transaction end indication at line 57 of column 11 of Tada constitutes the end transaction procedure. What this position ignores is the fact that a person of ordinary skill in the art looking to the teachings of Tada would recognize that everything that occurs after issuance of the transaction-end macro instruction (Tada 10:43-44) would constitute the end transaction procedure. Rather than concede that point, the Examiner now asserts that setting a transaction end indication (at the end of the end transaction procedure) constitutes the end transaction procedure. Although Tada teaches that the flushing of the transaction log occurs *during* the end transaction procedure, the Examiner ignores that particular fact and instead focuses on a very specific part of the end transaction, namely the setting of an indication to indicate that the transaction has ended. A person of ordinary skill in the art looking to the teachings of Tada would not read the Tada reference in the manner propounded by the Examiner. Rather, a person of ordinary skill in the art looking to the teachings of Tada would recognize that the log flushing is performed *during* the end transaction procedure, *not before* the end transaction procedure, as recited in claim 1.

Bohannon similarly teaches that the flushing of volatile tail 175 to non-volatile memory occurs during transaction *commit*, which is part of the end transaction procedure. *See* Bohannon, 10:4-9. Therefore, Bohannon is similar to Tada in teaching that the flushing of a transaction log occurs during, not before, an end transaction procedure. In view of the foregoing, even if Tada and Bohannon can be properly combined, the hypothetical combination does not teach or suggest each and every element of claim 1. Therefore, a *prima facie* case of obviousness has not been established with respect to claim 1.

Furthermore, there simply did not exist any motivation to modify the teachings of Tada to achieve the claimed invention of flushing a transaction log to non-volatile storage before an end transaction procedure. Both Tada and Bohannon suggest the complete opposite, namely, that flushing the transaction log is performed during an end transaction procedure. Thus, a person of ordinary skill in the art at the time of the invention, looking to the teachings of Tada and Bohannon, would have been led to performing transaction log flushing to non-volatile storage *during* end transaction processing, not before end transaction processing. Since, such a person of ordinary skill in the art would have been led away from the claimed invention, such a person would not have been motivated to achieve the claimed invention.

Therefore, because of a lack of motivation or suggestion to combine or modify the teachings of Tada and Bohannon to achieve the claimed invention, the *prima facie* case of obviousness is defective for this further reason.

Independent claims 17 and 24 are similarly allowable over the asserted combination of Tada and Bohannon.

For the foregoing reasons, it is respectfully requested the final rejection of the above claims be reversed.

2. Claims 4, 21 22, and 40.

Independent claim 21 is allowable for at least the same reasons as those given for independent claim 1, as the asserted combination of Tada and Bohannon does not teach or suggest flushing a transaction log from volatile storage to a non-volatile storage while the last step is performed by a plurality of access modules, where the last step is performed *before an end transaction procedure*.

Moreover, claim 21 recites determining that a last step of the transaction involves *the* plurality of access modules that processed a received transaction. The examiner cited to column 10, lines 35-37 of Tada as teaching the determining feature of claim 21. The cited passage of Tada refers to determining whether the processing at steps S02 through S04 has been carried out on all databases. There is no indication whatsoever in the cited passage of Tada that determining a last step of the transaction involves the plurality of access modules that process a received transaction.

Therefore, the hypothetical combination of Tada and Bohannon fails to teach or suggest this additional element of claim 21.

For the forgoing reasons, the reversal of the final rejection of the above claims is respectfully requested.

3. Claims 3, 6, 19, and 28.

Independent claim 28 recites a database system that has a parsing engine adapted to communicate a directive to access modules, with each access module responsive to the directive to perform a transaction log flush from the volatile storage to the persistent storage before performance of an end transaction procedure. As discussed above in connection with claim 1, the asserted combination of Tada and Bohannon clearly does not disclose or suggest performing a transaction log flush from volatile storage to persistent storage *before* performance of an end transaction procedure.

Claim 28 further recites that the parsing engine is adapted to avoid sending a broadcast directive to the access modules to cause performance of a transaction log flush *during* the end transaction procedure. This is directly contradicted by the teachings of both Tada and Bohannon, which teach that transaction log flushing occurs *during* an end transaction procedure.

In view of the foregoing, reversal of the final rejection is of the above claims is respectfully requested.

4. Claim 5.

Claim 5 depends from claim 1, and further recites determining if the transaction log has been flushed before performing the end transaction procedure. The Examiner cited Figure 5 and column 10, lines 39-47 of Tada as teaching this particular feature of claim 5. The cited passage of Tada refers to determining whether all databases have been processed, and if so, issuing a transaction-end macro instruction to start the end transaction procedure. There is absolutely no teaching whatsoever in the cited passage in column 10, or in Figure 5 of Tada, of the act of determining if transaction has been flushed *before performing* the end transaction procedure.

Consideration of whether the transaction log has been flushed before performing the end transaction procedure clearly is not a concern of Tada, where the transaction log *must* be flushed *during* the end transaction procedure. Bohannon similarly teaches that the transaction log has to be flushed during the end transaction procedure. Therefore, claim 5 is allowable over Tada and Bohannon for these additional reason.

Reversal of the final rejection of claim 5 is therefore respectfully requested.

5. Claim 7.

Claim 7, which depends from claim 1, further recites the act of identifying the transaction as an implicit transaction. The examiner conceded that neither Tada nor Bohannon explicitly discloses identifying the transaction as an implicit transaction. The basis for the obviousness rejection of claim 7 is that “[t]he database management system can use any available access method to access the database, which include a implicit transaction.” 4/5/2004 Office Action at 5-6. Such a rejection is based on speculation, as the examiner has not provided any objective evidence that would indicate a modification the teachings of Tada or Bohannon to enable the identification of a transaction as an implicit transaction, in combination with the other elements recited in claim 7.

In view of the foregoing, reversal of the rejection of claim 7 is respectfully requested.

6. Claims 9, 23, and 38.

Claim 9, which depends from claim 8 (which in turn depends from claim 1) recites that performing the end transaction procedure comprises skipping a broadcast of directive indicating commencement of the end transaction procedure to the plurality of access modules. This is clearly contradicted by the teaching of Tada, where an end transaction macro instruction (Tada,

10:43-44) is issued. Therefore, the hypothetical combination of Tada and Bohannon clearly does not teach *skipping* broadcast of a directive indicating commencement of the end transaction procedure to the plurality of users.

For the foregoing reasons, reversal of the final rejection of the above claims is respectfully requested.

7. Claim 30, 31 and 32, 41,42 and 43.

Claim 30 depends from claim 29, which in turn depends from claim 1. Claim 29 recites that the method further comprises performing the plural steps of the transaction *prior* to performing the end transaction procedure, where performing the flush of the transaction log is in one of the plural steps (prior to performing the end transaction procedure). Moreover, claim 30 recites performing, in each of the plural steps, access of relational table data stored in the database system. The examiner cited to step S13 of Figure 5 as teaching this feature of claim 30. 4/5/2004 Office Action at 12. Step S13 of Figure 5 occurs *after* the performance of the end transaction procedure, even the set transaction end indication (S12) identified by the Examiner in the Advisory Action. The requirement of claim 30 that the flushing of the transaction log occurs in one of the plural steps each accessing relational table data stored in the database system, where the plural steps are performed prior to the end transaction procedure, cannot be satisfied by step S13 of Figure 5 of Tada. Therefore, claim 30 is allowable over the asserted combination of Tada and Bohannon for this additional reason.

For the foregoing reasons, reversal of the final rejection of the above claims is respectfully requested.

8. Claim 33 and 39.

Claim 33 which depends from claim 4, (which in turn depends from claim 2) recites that performing the flush of the transaction is prior to the end transaction procedure if the last step is performed by all of the plurality of access modules. Claim 33 further recites that the method further comprises performing the flush of the transaction log in the end transaction procedure if the last step is *not performed by all of the plurality of access modules*. This determination, based on whether the last step is performed by all the plurality of access modules, for performing the transaction flush prior to or during the end transaction procedure is clearly not taught or suggested by either Tada or by Bohannon. In both Tada and Bohannon, the log flushing occurs only during an end transaction procedure -- no distinction is made based on whether a last step is performed by all of the plurality of access modules.

For the foregoing reasons, the reversal of the final rejection of the above claims is respectfully requested.

B. Claim 10-16 stand rejected under 35 U.S.C. § 103 over Tada alone.

1. Claims 10, and 12-16.

Independent claim 10 was rejected as being obvious over Tada alone. As conceded by the Examiner, Tada does not disclose a fallback module as recited in claim 10. However, the Examiner nevertheless stated that the subject matter of claim 10 would be obvious over Tada. It is respectfully submitted that the Examiner has failed to establish a *prima facie* case of obviousness with respect to claim 10, as there is no evidence of record that would suggest a modification of Tada to achieve the claimed invention.

Claim 10 recites a first access module in a database system writing an end transaction indication to a first transaction log portion, where the first access module is *part of a cluster of access modules*. Claim 10 also recites the first access module sending an end transaction directive to a *fallback module* associated with the first access module, where the fallback module is also part of the cluster. The Examiner stated that Tada somehow suggests the acts performed in claim 10, even though the Examiner conceded, with respect to claim 1, that "Tada does not explicitly disclose the transaction as processed by a plurality of access modules." 4/5/2004 Office Action at 4. In fact, in the rejection of claim 10, the Examiner made no mention whatsoever of the fact that claim 10 recites that the first access module is part of a cluster of *access modules* (note the plural sense of "access modules"). On at least this basis alone, the rejection of claim 10 is defective.

The Examiner then stated that "[f]allback is a redundancy operation in which a copy of a database portion is stored on a different access module than where the original data portion is stored. Tada teaches transferring data to buffer which a different storage location than where the original of the data portion is stored (Tada, col. 10, lines 29-67)." 4/5/2004 Office Action at 14-15. Although the cited column 10 passage of Tada describes issuing a transaction-end macro instruction, the transfer of log data to a log data buffer 132, and the transfer of the extracted log data to HLF buffer 114, there is absolutely no indication whatsoever of a first access module being part of a cluster of access modules, or a first access module sending an end transaction directive to a fallback module, where the fallback module is also part of the cluster.

The Examiner further stated that "[i]t is unclear to the Examiner what is a fallback module; therefore the Examiner interprets a fallback module as any processing module in a computer system." 4/5/2004 Office Action at 2. Appellant respectfully submits that the term

"fallback" is well defined in the specification. *See* Specification, page 15, line 17-page 18, line 7. The role of a "fallback access module" in accordance with an embodiment is also described on pages 15-18 of the Specification. Thus, the term "fallback module" is well supported in the Specification. Therefore, the term "fallback module" cannot be arbitrarily construed to mean any processing module. More importantly, by the Examiner's own admission, Tada does not disclose a transaction referenced by a plurality of access modules. If Tada does not disclose a plurality of access modules, then Tada clearly does not disclose a first access module and a fallback module as recited in the claim. In view of the foregoing, the obviousness rejection of claim 10 is defective.

For the foregoing reasons, the final rejection of claims 10 and 12-16 should be reversed.

2. Claim 11.

With respect to claim 11, the Examiner cited column 10, line 10-column 11, line 64, and Figure 5, of Tada as teaching the recited subject matter. However, if Tada does not teach plural access modules, then Tada clearly does not teach or suggest sending an end transaction directive to the fallback module but not to other access modules in a cluster. The rejection of claim 11 is clearly defective for this additional reason.

VIII. CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date:

9-20-04



Dan C. Hu, Reg. No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Ste. 100
Houston, TX 77024
713/468-8880 [Phone]
713/468-8883 [Fax]

APPENDIX OF CLAIMS

The claims on appeal are:

1. A method of performing a transaction in a database system, comprising:
receiving a transaction to be performed, wherein the transaction is processed by a plurality of access modules; and
performing a flush of a transaction log from volatile storage to non-volatile storage by each access module before an end transaction procedure.
2. The method of claim 1, further comprising issuing a request to flush the transaction log with a message sent to each access module for performing a last step of the transaction, the last step performed prior to the end transaction procedure.
3. The method of claim 2, further comprising performing the flush of the transaction log in a data access step prior to the end transaction procedure to avoid performance of a transaction log flush in the end transaction procedure.
4. The method of claim 2, further comprising determining that the last step is being performed by all of the plurality of access modules involved in the transaction.
5. The method of claim 1, further comprising determining if the transaction log has been flushed before performing the end transaction procedure.
6. The method of claim 5, further comprising avoiding performance of a transaction log flush in the end transaction procedure if the transaction log has been flushed.
7. The method of claim 1, further comprising:
identifying the transaction as an implicit transaction.

8. The method of claim 1, further comprising:
performing the end transaction procedure, which follows execution of the transaction.
9. The method of claim 8, performing the end transaction procedure comprising:
skipping broadcast of a directive indicating commencement of the end transaction procedure to the plurality of access modules.
10. A method of performing an end transaction procedure in a database system, comprising:
a first access module in the database system writing an end transaction indication to a first transaction log portion, the first access module being part of a cluster of access modules; and
the first access module sending an end transaction directive to a fallback module associated with the first access module, the fallback module being part of the cluster.
11. The method of claim 10, wherein the first access module sends the end transaction directive to the fallback module but not to other access modules in the cluster.
12. The method of claim 10, wherein sending the end transaction directive comprises sending an end transaction-part one directive.
13. The method of claim 12, further comprising the first access module broadcasting an end transaction-part two directive to all access modules in the cluster.

14. The method of claim 10, further comprising the fallback module writing an end transaction indication to a second transaction log portion.

15. The method of claim 10, further comprising the first access module flushing the first transaction log portion from volatile storage to non-volatile storage.

16. The method of claim 10, further comprising the first access module flushing the first transaction log portions but the other access modules in the cluster not flushing their respective transaction log portions.

17. A database system comprising:
a plurality of storage media, the storage media comprising persistent storage;
volatile storage; and
a plurality of access modules, wherein each access module is coupled to one of the plurality of storage media; and
each of the access modules being adapted to flush a transaction log from the volatile storage to the persistent storage before performing an end transaction procedure.

18. The database system of claim 17, further comprising a controller adapted to determine if each access module has flushed the transaction log maintained by the access module.

19. The database system of claim 18, wherein the controller is adapted to skip sending a directive to perform a transaction log flush if the controller determines that each access module has flushed the transaction log before the end transaction procedure.

20. The database system of claim 17, further comprising a controller adapted to provide a flush directive with a message to each of the access modules to perform a last step of the transaction before the end transaction procedure.

21. An article comprising a medium storing instructions for enabling a processor-based system to:

receive a transaction to be performed, wherein the transaction is processed by a plurality of access modules;

determine that a last step of the transaction involves the plurality of access modules, wherein the last step is performed before an end transaction procedure; and

flush a transaction log from volatile storage to a non-volatile storage while the last step is performed by the plurality of access modules.

22. The article of claim 21, further storing instructions for enabling the processor-based system to:

perform the end transaction procedure, wherein the end transaction procedure follows execution of the last step of the transaction.

23. The article of claim 22, further storing instructions for enabling a processor-based system to:

avoid broadcast of a directive indicating commencement of the end transaction procedure to the plurality of access modules.

24. A method of performing a transaction in a database system, comprising:
receiving a transaction to be performed on plural access modules in the database system;

maintaining a log in volatile storage to track operations performed in the transaction; and

writing the log to persistent storage before start of an end transaction procedure.

25. The method of claim 24, wherein writing the log to persistent storage comprises flushing the log.

26. The method of claim 24, wherein maintaining the log comprises maintaining a transaction log.

27. The method of claim 24, further comprising performing the end transaction procedure, the end transaction procedure comprising writing an end transaction indication into the log.

28. A database system comprising:
storage media comprising persistent storage;
volatile storage;
access modules coupled to the storage media; and
a parsing engine coupled to the access modules, the parsing engine adapted to perform one of:

(a) providing a directive with a message to perform a last step of a transaction and communicating the directive to the access modules, each access module responsive to the directive to perform a transaction log flush from the volatile storage to the persistent storage before performance of an end transaction procedure; and

(b) determining if each of the access modules has performed a transaction log flush before start of the end transaction procedure;
the parsing engine adapted to avoid sending a broadcast directive to the access modules to cause performance of a transaction log flush during the end transaction procedure.

29. The method of claim 1, wherein the transaction comprises plural steps, the method further comprising:

performing the plural steps prior to performing the end transaction procedure, and
wherein performing the flush of the transaction log comprises performing the flush of the transaction log in one of the plural steps.

30. The method of claim 29, wherein performing the plural steps comprises performing, in each of the plural steps, access of relational table data stored in the database system.

31. The method of claim 30, wherein performing the flush of the transaction log in one of the plural steps comprises performing the flush of the transaction log in a last one of the plural steps.

32. The method of claim 31, further comprising each access module adding a first entry to the transaction log to redo the transaction by the access module in case of system failure.

33. The method of claim 4, wherein performing the flush of the transaction is prior to the end transaction procedure if the last step is performed by all of the plurality of access modules, the method further comprising:

performing the flush of the transaction log in the end transaction procedure if the last step is not performed by all of the plurality of access modules.

34. The database system of claim 17, wherein the access modules to perform a transaction comprising plural steps, one or more of the access modules adapted to perform the plural steps prior to the end transaction procedure, and the access modules adapted to perform the flush of the transaction log in one of the plural steps.

35. The database system of claim 34, wherein the one of the plural steps comprises a last one of the steps.

36. The database system of claim 35, wherein the transaction log comprises a first entry associated with each access module to enable a redo of the transaction in case of system failure.

37. The database system of claim 36, wherein the transaction log further comprises a second entry associated with each access module to enable an undo of the transaction.

38. The database system of claim 34, further comprising a controller to determine whether a last one of the steps involves all the access modules, and in response to determining

that the last one of the steps involves all the access modules, the controller to send a directive to all the access modules to perform the flush of the transaction log in the last one of the steps.

39. The database system of claim 38, in response to determining that the last step does not involve all access modules, the controller to send a directive to perform the flush of the transaction log in the end transaction procedure.

40. The article of claim 21, wherein the transaction comprises plural steps, the article further storing instructions for enabling a processor-based system to:
perform the plural steps prior to performing the end transaction procedure, and
wherein performing the flush of the transaction log comprises performing the flush of the transaction log in one of the plural steps.

41. The article of claim 40, wherein performing the plural steps comprises performing, in each of the plural steps, access of relational table data stored in the database system.

42. The article of claim 41, wherein performing the flush of the transaction log in one of the plural steps comprises performing the flush of the transaction log in a last one of the plural steps.

43. The article of claim 42, further storing instructions for enabling a processor-based system to cause each access module to add a first entry to the transaction log to redo the transaction by the access module in case of system failure.